


Unidade VII: Árvore Binária - Introdução



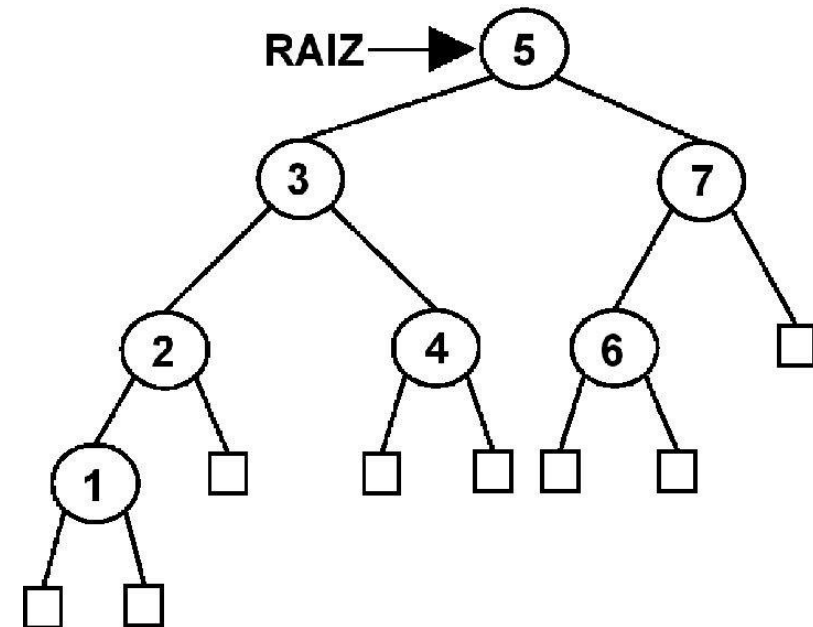
PUC Minas

Adaptação dos slides elaborados pelo Instituto de Ciências Exatas e
Informática - Departamento de Ciência da Computação

- Definições e conceitos
- Classe Nó
- Classe Árvore Binária

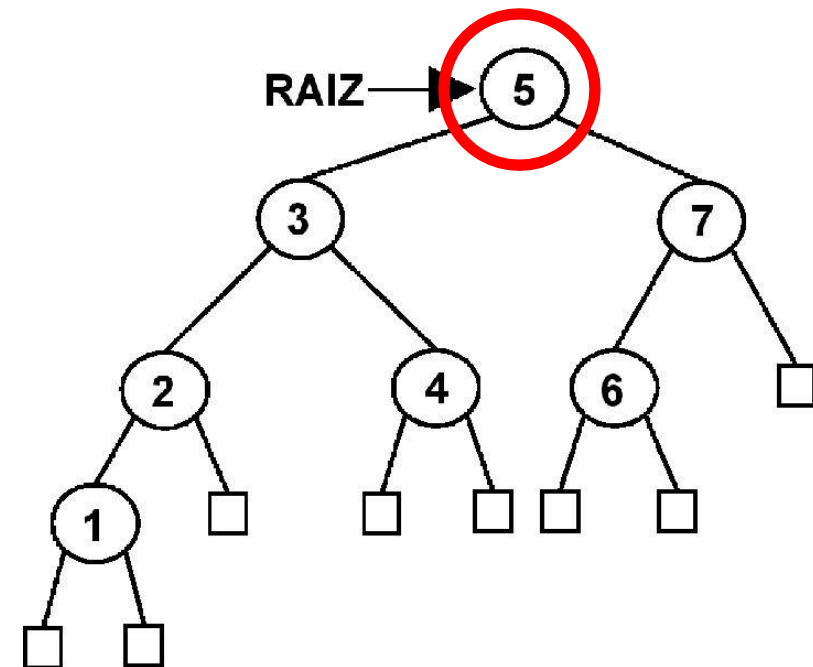
- **Definições e conceitos** 
- Classe Nó
- Classe Árvore Binária

- Custo de inserção, remoção e pesquisa **pode** ser $\Theta(\lg(n))$ comparações
- Formada por um conjunto finito de nós (vértices) conectados por arestas



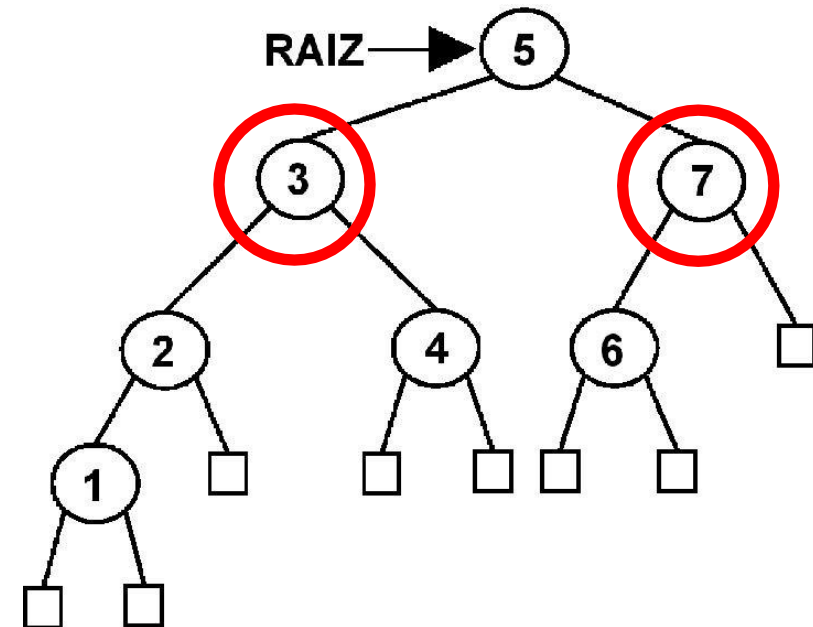
- Custo de inserção, remoção e pesquisa pode ser $\Theta(\lg(n))$ comparações
- Formada por um conjunto finito de nós (vértices) conectados por arestas

O nó 5 é denominado nó raiz e ele está no nível 0



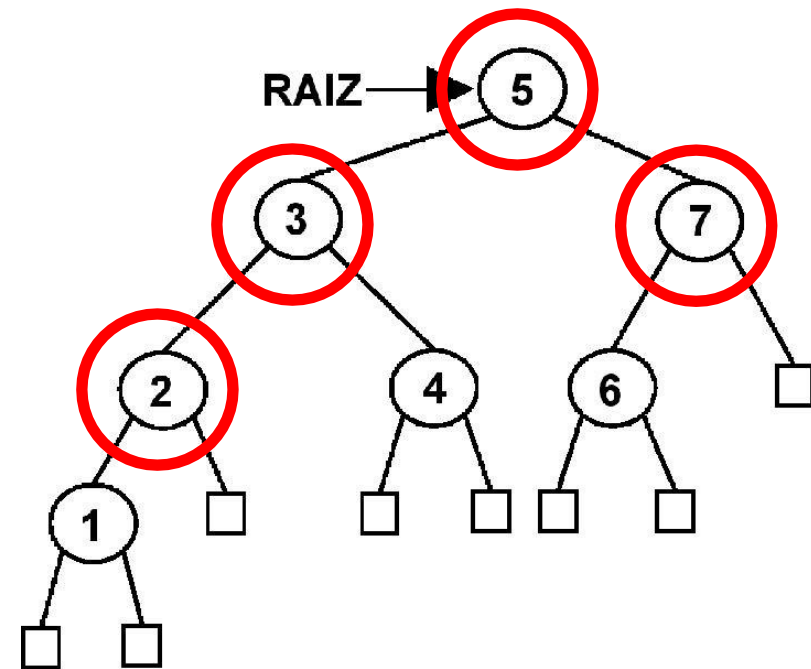
- Custo de inserção, remoção e pesquisa pode ser $\Theta(\lg(n))$ comparações
- Formada por um conjunto finito de nós (vértices) conectados por arestas

Os nós 3 e 7 são filhos do 5.
O nó 5 é pai dos nós 3 e 7.



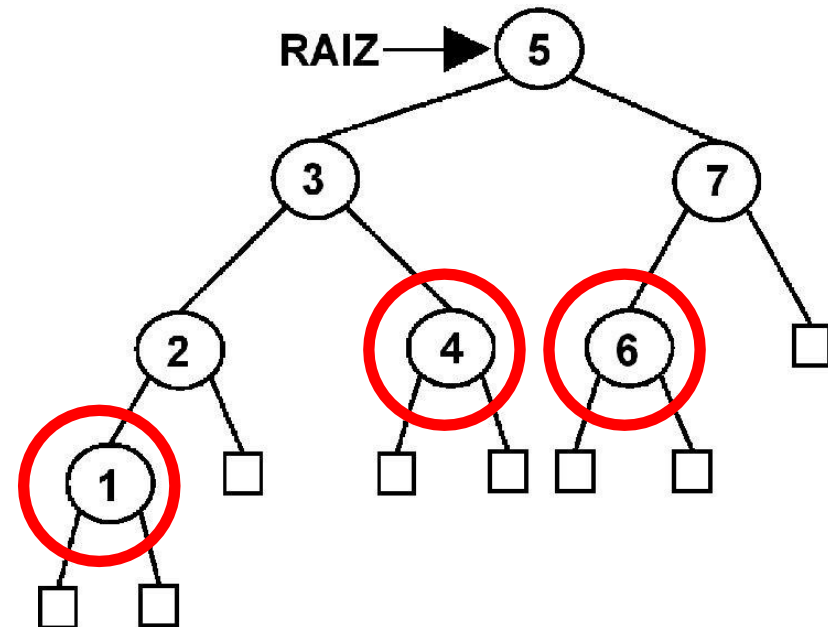
- Custo de inserção, remoção e pesquisa pode ser $\Theta(\lg(n))$ comparações
- Formada por um conjunto finito de nós (vértices) conectados por arestas

Um nó com filho(s) é chamado de **nó interno**.



- Custo de inserção, remoção e pesquisa pode ser $\Theta(\lg(n))$ comparações
- Formada por um conjunto finito de nós (vértices) conectados por arestas

Um nó sem filhos é chamado de
nó **folha**.

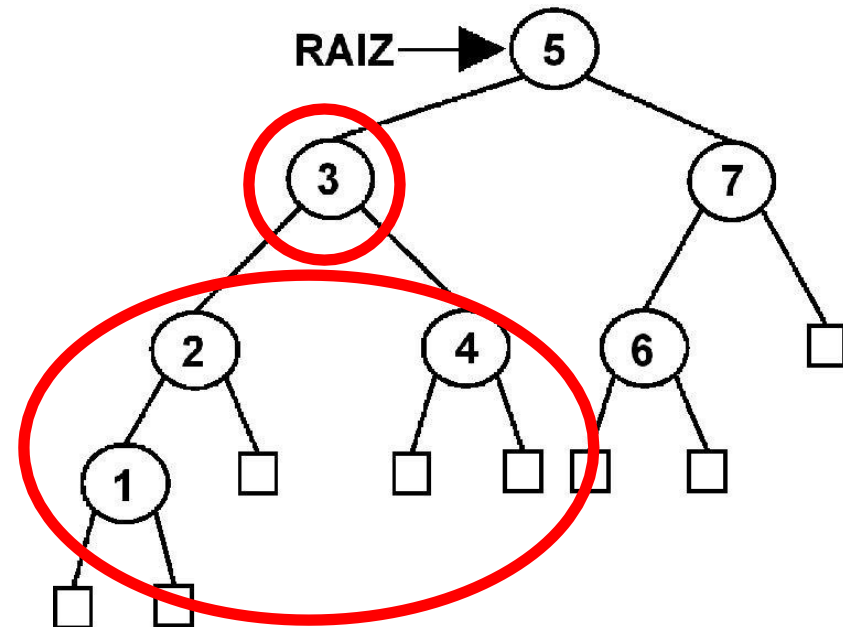


Descobriram Nossa Árvore



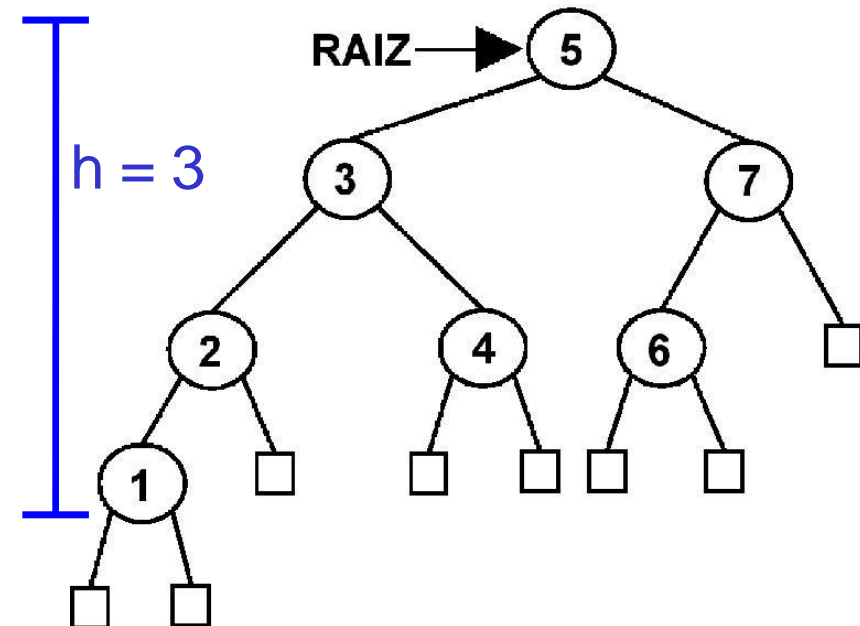
- Custo de inserção, remoção e pesquisa pode ser $\Theta(\lg(n))$ comparações
- Formada por um conjunto finito de nós (vértices) conectados por arestas

Os nós 1, 2 e 4 formam uma subárvore com raiz no nó 3



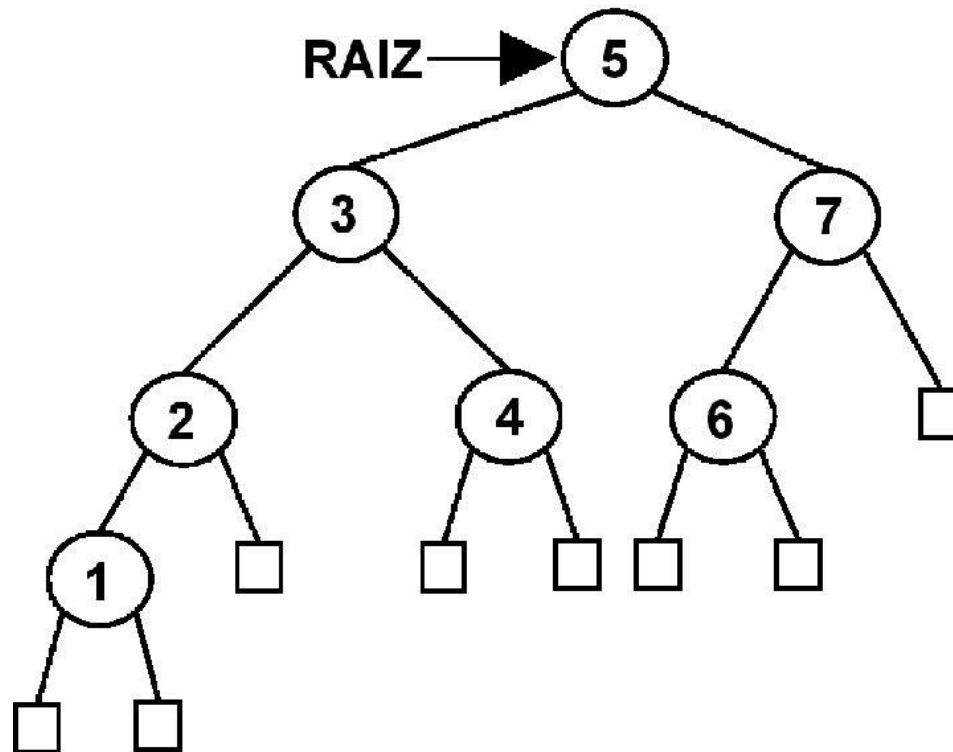
- Custo de inserção, remoção e pesquisa pode ser $\Theta(\lg(n))$ comparações
- Formada por um conjunto finito de nós (vértices) conectados por arestas

Altura (h): maior distância entre um nó e a raiz



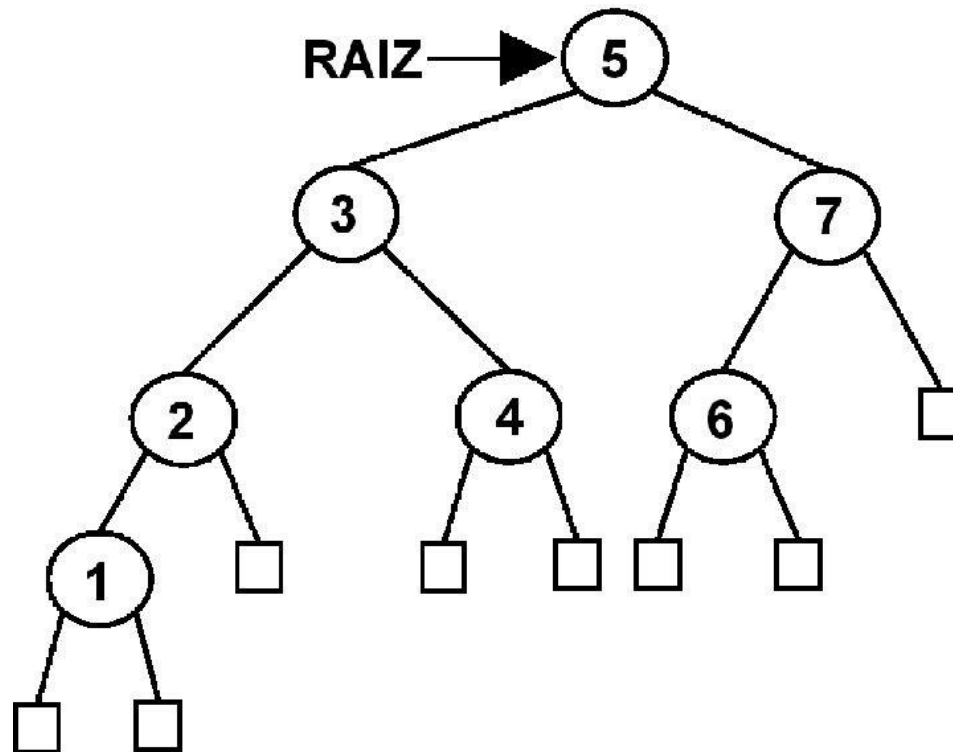
Árvore Binária

- Árvore em que cada nó possui **no máximo dois filhos**, por exemplo:



Árvore Binária de Pesquisa

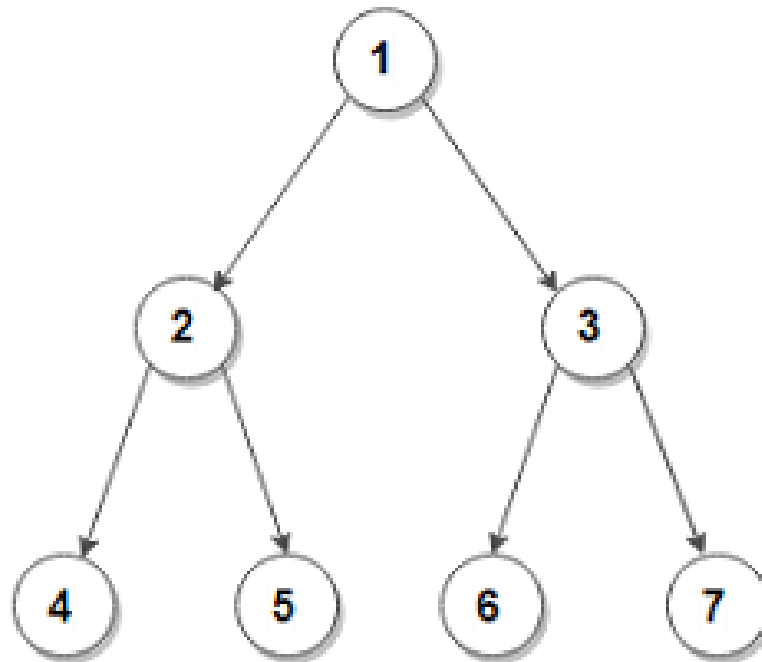
- Árvore binária em que cada nó é maior que todos seus vizinhos à esquerda e menor que todos à direita. Por exemplo:



Árvore Binária Completa

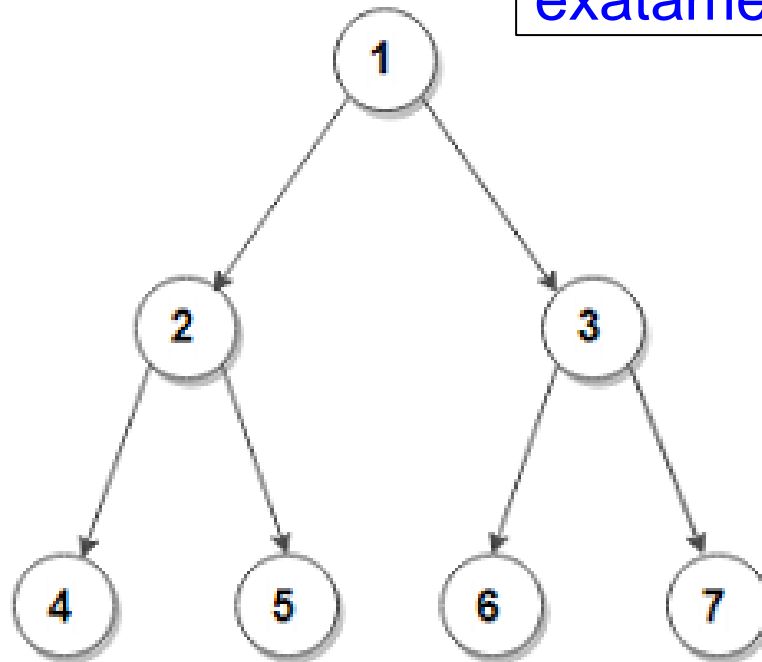
- Árvore binária em que:
 - Cada nó é uma folha **OU** possui exatamente dois filhos
 - Todos os nós folhas possuem uma altura h
 - O número de nós internos é $2^h - 1$
 - O número de nós folhas é 2^h
 - O número total de nós é $(2^h - 1) + 2^h = 2^{(h+1)} - 1$

Exemplo- Árvore binária completa

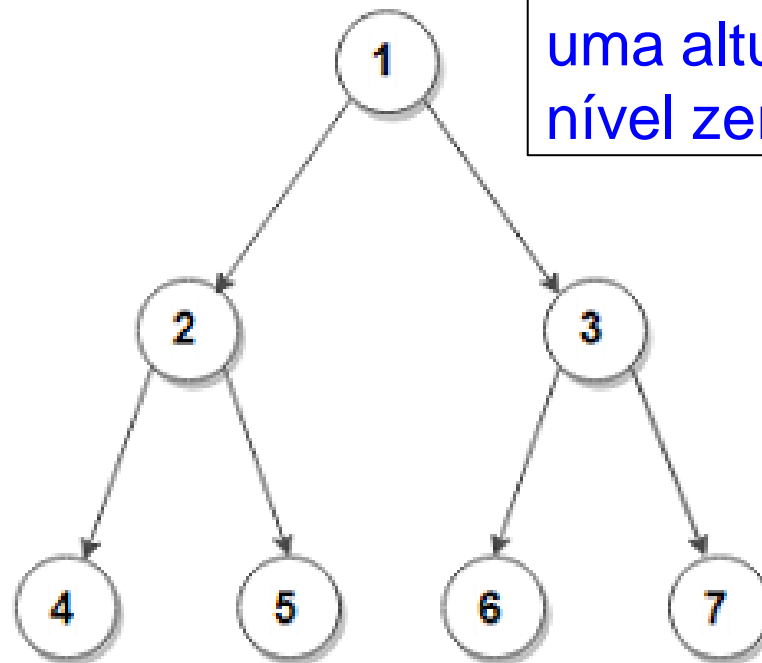


Exemplo- Árvore binária completa

Cada nó é uma folha **OU** possui exatamente dois filhos



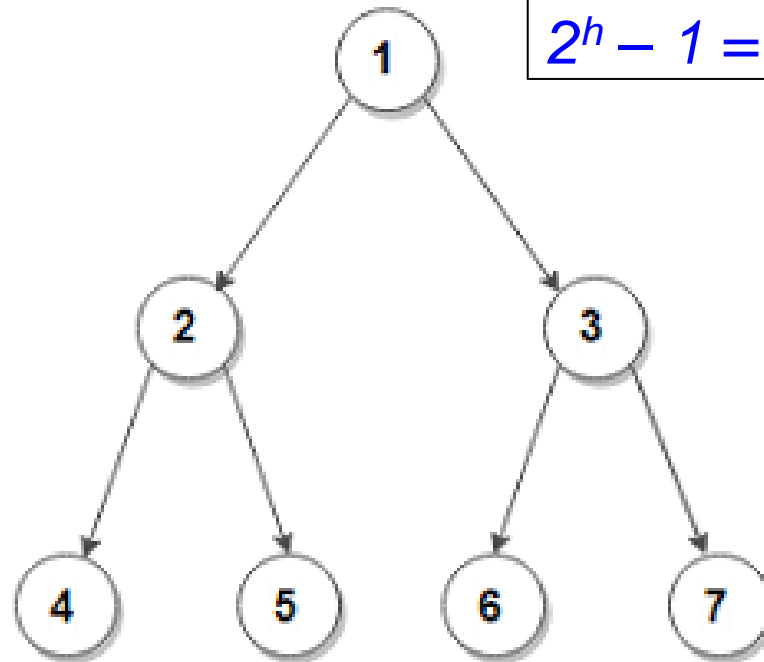
Exemplo- Árvore binária completa



Todos os nós folhas possuem uma altura $h = 2$ (a raiz está no nível zero)



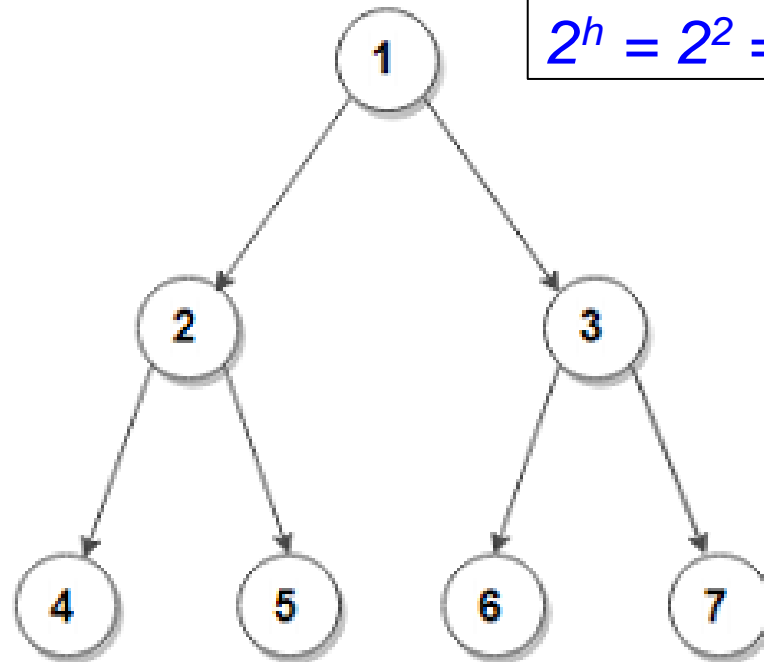
Exemplo- Árvore binária completa



O número de nós internos é
 $2^h - 1 = 2^2 - 1 = 3$



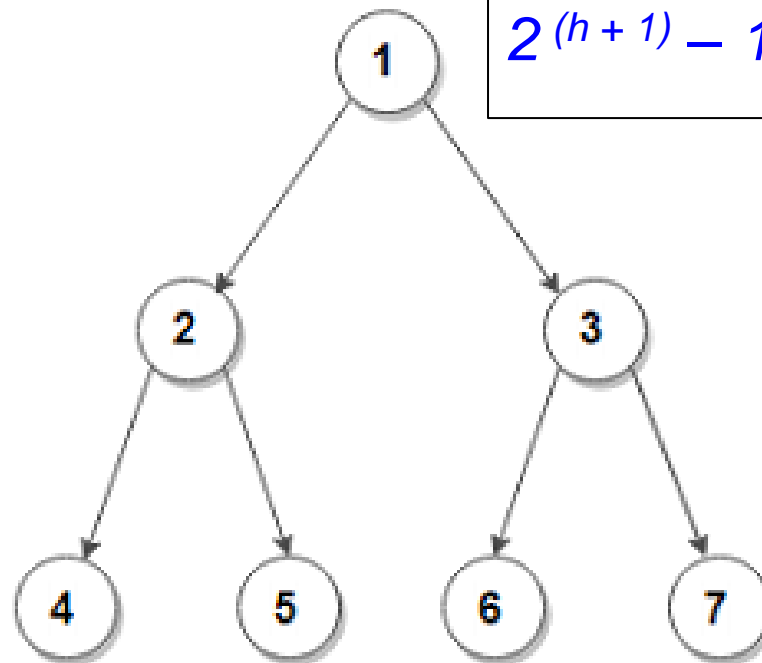
Exemplo- Árvore binária completa



O número de nós folhas é
 $2^h = 2^2 = 4$



Exemplo- Árvore binária completa

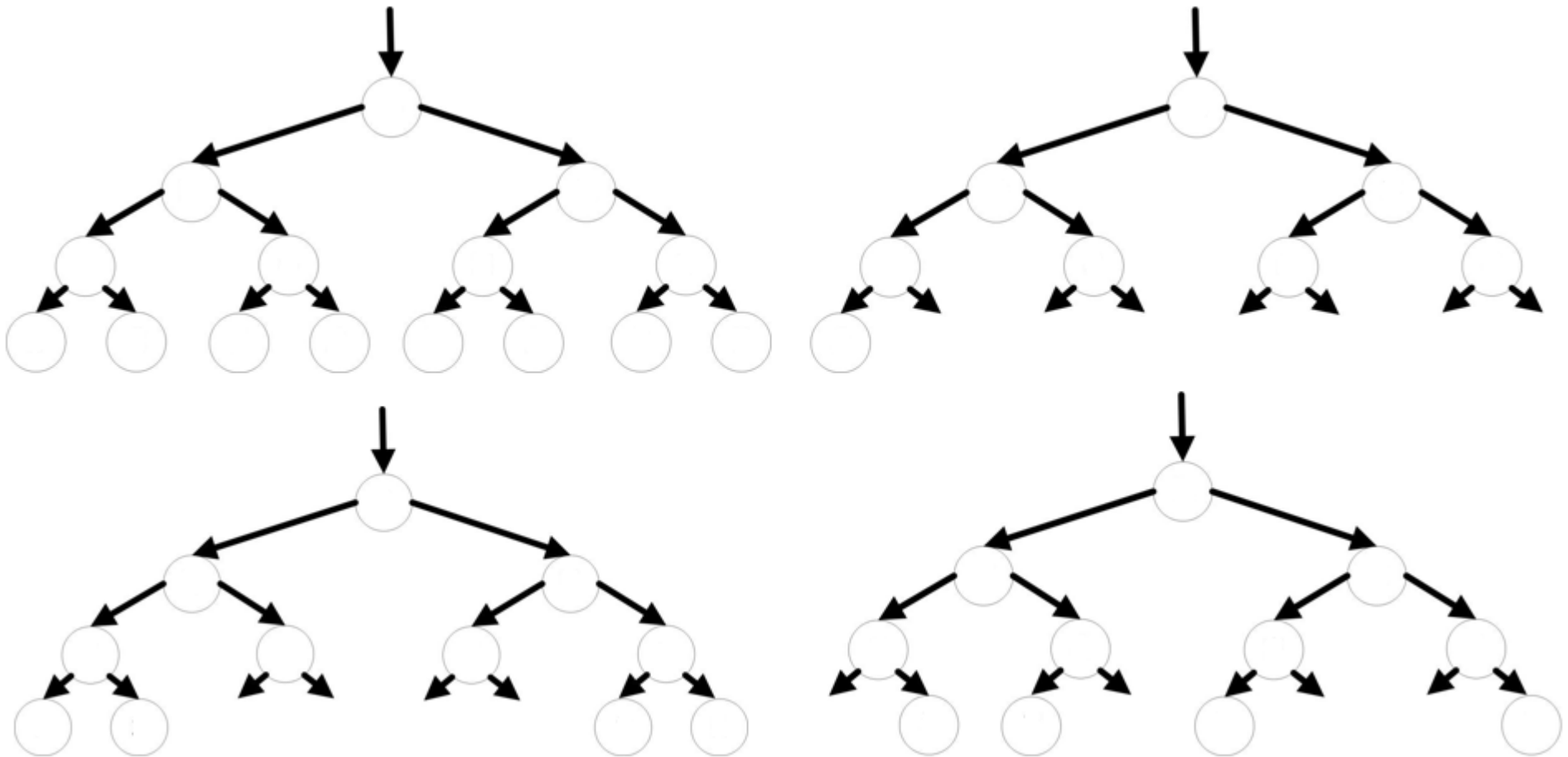


O número total de nós é
 $2^{(h+1)} - 1 = 2^{(2+1)} - 1 = 7$



Árvore Balanceada

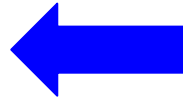
- Árvore em que para **TODOS** os nós, a diferença entre a altura de suas árvores da esquerda e da direita sempre será **0** ou **± 1** como, por exemplo:



Consideração

- Árvore Binária de Pesquisa (ABP) também é chamada de:
 - Árvore Binária de Busca (ABB) ou
 - Binary Search Tree (BST)
- A partir deste ponto, neste material, considera-se que todas as árvores binárias serão de pesquisa

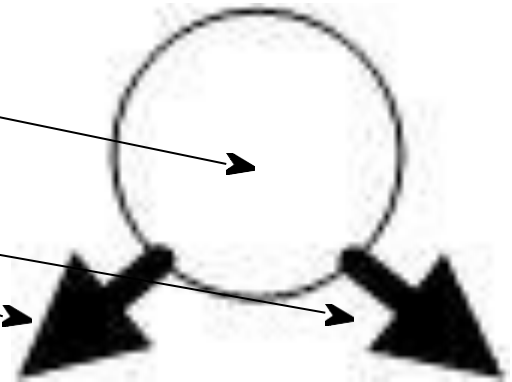
- Definições e conceitos
- **Classe Nó**
- Classe Árvore Binária



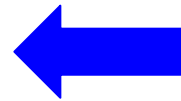
Algoritmo - Classe Nó

```
class No
{
    private int elemento;
    private No esq;
    private No dir;
    public No(int elemento)
    {
        this.elemento = elemento;
        esq = null;
        dir = null;
    }
    public No(int elemento, No esq, No dir)
    {
        this.elemento = elemento;
        this.esq = esq;
        this.dir = dir;
    }

    ...
    //Incluir as Propriedades
}
```



- Definições e conceitos
- Classe Nó
- **Classe Árvore Binária**



Classe Árvore Binária

```
class ArvoreBinaria
{
    private No raiz;
    public ArvoreBinaria(){raiz = null;}
    public void Inserir(int x) { }
    public bool Pesquisar(int x) { }
    public void CaminharCentral() { }
    public void CaminharPre() { }
    public void CaminharPos() { }
    public void Remover(int x) { }
}
```

Classe Árvore Binária

```
class ArvoreBinaria
{
    private No raiz;
    public ArvoreBinaria(){raiz = null;}
    public void Inserir(int x) { }
    public bool Pesquisar(int x) { }
    public void CaminharCentral() { }
    public void CaminharPre() { }
    public void CaminharPos() { }
    public void Remover(int x) { }
}
```

raiz



Raiz

